

Long Concept Query on Conceptual Taxonomies

Yi Zhang^{*}, Yanghua Xiao[†], Seung-won Hwang[‡], Wei Wang[§]

^{*} [†] Fudan University, Shanghai, China

[‡] Yonsei University, Seoul, Korea

[§] University of California, Los Angeles, CA, USA

^{*}[†] {z_yi11, shawyh}@fudan.edu.cn, [‡]seungwonh@yonsei.ac.kr, [§]weiwang@cs.ucla.edu

Abstract

This paper studies the problem of finding typical entities when the concept is given as a query. For a short concept such as *university*, this is a well-studied problem of retrieving *knowledge base* such as Microsoft’s Probase and Google’s isA database pre-materializing entities found for the concept in Hearst patterns of the web corpus. However, we find most real-life queries are long concept queries (LCQs), such as *top American private university*, which cannot and should not be pre-materialized. Our goal is an online construction of entity retrieval for LCQs. We argue a naive baseline of rewriting LCQs into an intersection of an expanded set of composing short concepts leads to highly precise results with extremely low recall. Instead, we propose to augment the concept list, by identifying related concepts of the query concept. However, as such increase of recall often invites false positives and decreases precision in return, we propose the following two techniques: First, we identify concepts with different relatedness to generate linear orderings and pairwise ordering constraints. Second, we rank entities trying to avoid conflicts with these constraints, to prune out lowly ranked one (likely false positives). With these novel techniques, our approach significantly outperforms state-of-the-arts.

Introduction

We study the problem of returning entities for the name of a concept as a query. For example, a user may query *top American private university* to obtain the most typical examples such as {*Harvard*, *Stanford*, *Yale*} as results.

One related effort is retrieving *knowledge base* materializing isA relationships between entity-concept pairs, such as Microsoft’s Probase and Google’s isA database. These knowledge bases are constructed by extracting Hearst patterns from web corpus, such as ‘American universities such as Harvard and Yale’. However, Figure 1 shows the distribution of # of modifiers used in the concepts of Probase, indicating that more than 60% of concepts have zero or one modifier. In other words, this resource cannot answer long concept query (LCQ) such as *top American private university*, which is dominant in search engine queries. A statistical study reveals that LCQ accounts for a majority of searches on search engine: based on a one week worth of search log (from 07/25/2012 to 07/31/2012) of BING (Wang, Wang, and Hu 2014), about 90% of distinct queries have one or

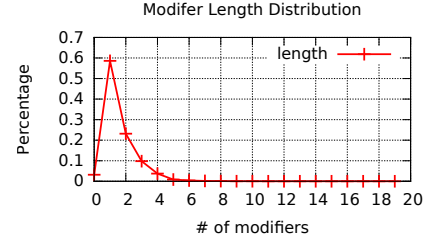


Figure 1: Modifier length distribution in Probase

more modifiers which differs significantly from Figure 1. Our goal is to bridge such a gap, by an online materialization for LCQs, using the materialized concepts in Probase.¹

Another related effort is *set expansion*. In this problem, a small set of seed entities are given as a query, such as {*Harvard*, *Yale*}, to obtain more instances such as *Stanford*. As we empirically show later, our solution can solve this task, by identifying related concepts to the query and return the typical examples, but we cannot use existing set expansion solutions for our problem, as they require example entities which directly reveal the query intent as input. Existing work on set expansion has many applications— a) knowledge base expansion to mine more instances and b) automatic list completion in spreadsheet. Our work, by automatically finding possible sets of seed entities, can apply to the above and more, enabling also the completion of the concept/list with few or even no examples.

Toward this goal, we first describe a high-precision baseline and discuss the challenges in improving recall while maintaining precision.

High-precision Baseline A naive baseline is decomposing the given long concept c_q , e.g., *top American private university*, into a set of composing shorter concepts C_q , e.g., {*top university*, *American university*, *private university*}. We can intersect the instances of $c \in C_q$. However, as knowledge bases are extracted from text patterns demonstrating a few examples (and not aiming to enumerate all instances), the recall of instances in each concept is low and this problem exacerbates as more concepts are intersected. For example, the concept *low-fertility country* in Probase contains only 4 entities: {*Japan*, *Germany*, *Spain*, *Canada*}. However,

¹Note we do not prematerialize all LCQs, as materialization of concepts of length n generates 2^{n-1} combinations.

none of them is among the top 20 lowest fertility countries in 2014. Many typical instances in this concept such as *Singapore* and *South Korea* are missing due to the rare mentions by Hearst patterns.

We thus propose our approach to solve this problem and the main idea is to first, increase the recall by introducing new concepts to C_q . For example, adding *Ivy League*, which is nearly equivalent to the query concept *top American private university*, to C_q would significantly increase the recall. However, some of such addition of a concept may introduce false positive instances, for which we devise ranking for pruning.

Recall We have two basic ideas to improve the recall, corresponding to two situations for the instance intersection of shorter concepts: $\cap_{c \in C_q} e(c)$ is empty or not, where $e(c)$ is the instances of c found from the knowledge base. When it is not empty, which implies that we have found a certain number of seed instances. These seeds are strong signals for us to find more related concepts and in turn to find more result entities. However, in most cases, $\cap_{c \in C_q} e(c)$ is empty due to the sparsity of current knowledge bases. In these cases, we find more sets of seed instances by relaxing the constraint on the seed instances. Specifically, instead of using $\cup_c \in C_q$ as seeds, we use $\cap_{c \in C'_q} e(c)$ such that $C'_q \subset C_q$, i.e., the *instance intersection of subsets of shorter concepts*. Since we intersect less concepts, it is quite possible to find a C'_q whose intersection is not empty. Given non-empty intersections, we use them as seed and expand the concept in the same way as the previous case.

Precision After we get all candidate entities, we still need to rank them together. In the concept expansion step, we have derived a linear ordering and pairwise ordering constraints. We need to aggregate all these information including an entity ordering derived without introducing any expanded concept to generate a better global ranking with fewer conflicts. Thus, we propose a new aggregation method based on a learning-to-rank framework to solve this problem, combined with both global and pairwise constraints.

To conclude, our problem is that given a long query concept c_q , return its top- k entities which are the k most typical instances of c_q . Our framework is as follows. First, generate an entity ranking list for the query concept without introducing some other concepts by an iterative method. It provides us a baseline ranking for entities bias to the precision which is an important global ordering constraint. Second, we propose to find possible seed instances for the LCQ, and propose an instance-based concept expansion to improve the recall. Here, we expand concepts by two probabilistic models, Noisy-Or model and Naive-Bayes model. Third, we generate two kinds of constraints based on the baseline ranking and the concepts expanded. A new entity ranking aggregation method is proposed to minimize the conflicts of the constraints. The new ranking can maintain a high precision while improving the recall.

Related Work

Retrieving typical instances of the given concept was studied, often in the context of building a knowledge base. Microsoft's Probase and Google's isA database mine textual

patterns, such as Hearst patterns, to collect isA relations of concepts. However, long concepts are sparsely observed in the text and thus cannot be obtained in this manner. Our problem distinguishes itself for being *online process* of mining instances for LCQ. We consider baselines of leveraging existing knowledge bases materializing short concepts: First, we can decompose a LCQ into composing short concepts and intersect the instances, which leads to highly precise but very sparse results. Second, the above set can be expanded by *set expansion* work, which we describe below.

Set expansion considers that a small set of seed entities are given as a query, such as $\{\textit{Harvard}, \textit{Yale}\}$, to obtain more instances. The approach is to take the collection (seed entities) as input to find related entities to populate the input set.

Google Sets is a product implementation used to populate a spreadsheet after users provide some instances as examples. Inspired by Google Sets, many research works followed (Ghahramani and Heller ; He and Xin 2011; Wang and Cohen 2008; Sarmiento et al. 2007; Pantel et al. 2009), to measure the membership strength of an item to the hidden concepts exemplified by the query entities.

Related problems include harvesting lists on the Web, and retrieving the list that completes the seed instances (He and Xin 2011; Wang et al. 2015). Compared with these works, our task is more challenging relying solely on entities and concepts in the knowledge base without rich exclusive lists from the Web. Most of the entity lists of the concept in the knowledge base are incomplete and inaccurate which makes the inference much more difficult.

However, this line of approaches may lead to ambiguity when given a long concept query. For example, when we get seed entities $\{\textit{Yale}, \textit{Harvard}, \textit{Stanford}\}$ as input from the decomposition of the concept *American private university*, we lose the information that the results we need should not only belong to American universities but also private universities. Set expansion approaches tend to mix entities belonging to different concepts during expansion which will lead to a lot of false-positives.

Our proposed solution can complement both knowledge base construction and set expansion. For knowledge base, we can use our solution to mine instances for the missing concepts or grow the instance set of the concept that is too sparse. For set expansion, our approach allows to consider instance lists, by aggregating them with the awareness of the semantic relationships of concepts.

Another line of related work is to aggregate different rankings. It is also named as Kemeny rank aggregation (Ali and Meilă 2012; Betzler, Bredebeck, and Niedermeier 2014; Betzler et al. 2010; Conitzer, Davenport, and Kalagnanam 2006), consensus ranking (Meila et al. 2012), median ranking (Cohen-Boulakia, Denise, and Hamel 2011), and preference aggregation (Davenport and Kalagnanam). These ranking aggregation methods have been of particular interest in the information retrieval and database communities. All these works are defined on a set of permutations. However, in our problem, we not only need to aggregate linear orderings (permutations) but also some pairwise ordering constraints which are not permutations.

Baseline Ranking

In this section, we present our baseline ranking model. This model underlies that our further development in the next two sections.

The baseline ranking is based on the following mutually recursive principles:

Principle 1 *An entity e is unlikely to be the answer, if it is related to none of the concepts in \mathbf{C}_q .*

Principle 2 *A concept c in \mathbf{C}_q is not informative, if it contains none of the entities which are likely to be the answer.*

The rationality of the principles are as follows. Due to the data sparsity of the knowledge base, an entity is not very likely to be related to every concept in \mathbf{C}_q even it is one of the answers. We assume that only if the entity is not related to any of the concept in \mathbf{C}_q , then the entity is not the answer. Hence, based on our principles, an answer entity does not have to be related to every concept to get a high probability. Similarly, the concept is *informative* (promising to be the concept whose entities are the answers) if it contains at least one entity which is very likely to be the answer. In general, the more important entities the concept contains, the more informative the concept is.

Clearly, the Noisy-Or model can reflect the above principles and rationalities. Here, we use $\sigma(e)$ to represent the probability of the entity e to be the answer, and $\sigma(c)$ to represent the informativeness of the concept c . We have:

$$\sigma(e) = 1 - \prod_{c \in c(e), c \in \mathbf{C}_q} (1 - \sigma(c)) \quad (1)$$

$$\sigma(c) = 1 - \prod_{e \in e(c)} (1 - \sigma(e)) \quad (2)$$

where $c(e)$ is the concepts of e found from the knowledge base.

In Eq. 1 and Eq. 2, $1 - \sigma(c)$ and $1 - \sigma(e)$ are usually quite small and multiplying many of them may lead to underflow. To facilitate computation, we use logarithm to transfer the equations into their equivalent logarithmic forms:

$$\log(1 - \sigma(e)) = \sum_{c \in c(e), c \in \mathbf{C}_q} \log(1 - \sigma(c)) \quad (3)$$

$$\log(1 - \sigma(c)) = \sum_{e \in e(c)} \log(1 - \sigma(e)) \quad (4)$$

As in Authority-hub analysis (Kleinberg 1999) and PageRank (Page et al. 1999), we adopt an iterative method to compute $\sigma(c)$ and $\sigma(e)$. We choose the initial state in which all concepts $c \in \mathbf{C}_q$ have a uniform probability to be an informative concept. In each iteration, our algorithm first uses $\sigma(c)$ to compute $\sigma(e)$, and then recomputes $\sigma(c)$ based on the $\sigma(e)$. The iteration stops when it reaches a stable state where for each $e \in E$, $\sigma(e)$ changes a little after an iteration (Yin, Han, and Yu 2008).

Concept Expansion

In this section, we introduce how we improve the recall by finding more related concepts. We first show how we expand concept by instances when $E(\mathbf{C}_q) = \bigcap_{c \in \mathbf{C}_q} e(c) \neq \emptyset$. For the cases where $E(\mathbf{C}_q) = \emptyset$, we find a subset \mathbf{C}'_q of \mathbf{C}_q so that its instance intersection is not empty.

Instance-based Expansion

We first present our principle for instance-based concept expansion in Principle 3 and illustrate it in Example 1. There are two conditions in the principle. Next, we elaborate each one.

In the first condition, we claim that a strong signal of semantic equivalence or relatedness between concept pairs is their overlap of respective instances. Thus, if a concept's entities contain most of the entities in $\bigcap_{c \in \mathbf{C}_q} e(c)$, it is very likely that the concept is a closely related concept of c_q . As shown in Example 1, the concept *Ivy League* contains almost all instances in $\bigcap_{i=1}^3 e(c_i)$.

Principle 3 *A concept c is semantically related to c_q , if (1) most instances in $\bigcap_{c \in \mathbf{C}_q} e(c)$ are related to c and (2) c has as few as possible instances $e \notin \bigcup_{c \in \mathbf{C}_q} e(c)$*

Example 1 *Suppose c_q : top American private university, then $\mathbf{C}_q = \{c_1: \text{top university}, c_2: \text{American university}, c_3: \text{private university}\}$. In Probase, we have $\bigcap_{i=1}^3 e(c_i) = \{\text{harvard, princeton, yale, pennsylvania state university, cornell}\}$. Clearly, 'Ivy League' is a closely related concept of c_q . In Probase, it contains most of the entities in $\bigcap_{i=1}^3 e(c_i)$ and few entities not in $\bigcup_{i=1}^3 e(c_i)$. Hence, we can use instances in Ivy League such as $\{\text{dartmouth, columbia}\}$ to improve the recall.*

By the second condition, we hope to exclude an over-general concept. Theoretically, if a concept contains many entities not related to any shorter concept of c_q , it is not a good concept to expand. In our running example, another concept *famous university* contains all entities in $\bigcap_{i=1}^3 e(c_i)$. However, it also contains entities such as *loughborough*, which is not a member of any of shorter concepts. Thus, *famous university* should not be considered.

Formally, we define a relevance score $rel(c, \mathbf{C}_q)$ to rank the concepts expanded which reveals that how likely c is to be a semantically related concept of c_q . Clearly, the Naive-Bayes model can reflect our principle. The model is as follows:

$$\begin{aligned} rel(c, \mathbf{C}_q) &= \frac{P(c|E(\mathbf{C}_q))}{g(c)} \propto \frac{P(E(\mathbf{C}_q)|c)P(c)}{g(c)} \\ &= \frac{P(c) \prod_{e \in E(\mathbf{C}_q)} P(e|c)}{g(c)} \end{aligned} \quad (5)$$

In the nominator, $\prod_{e \in E(\mathbf{C}_q)} P(e|c)$ implies that only when most instances in $E(\mathbf{C}_q)$ are related to c , c will get a high ranking score.

Generally, there are relatively few concepts related to all of the entities in the intersection due to the sparsity in the current knowledge base. Therefore, appropriate smoothing is necessary to avoid zero probabilities. To do this, we can assume that with probability $(1 - \gamma)$, the concept should be chosen by its prior typicality.

$$= \frac{P(c) \prod_{e \in E(\mathbf{C}_q)} (\gamma P(e|c) + (1 - \gamma)P(e))}{g(c)} \quad (6)$$

Another option to attack the sparsity problem is relaxing the first condition in the Principle 3 by a Noisy-Or model:

$$rel(c, \mathbf{C}_q) = \frac{P(c|E(\mathbf{C}_q))}{g(c)} = \frac{1 - (1 - \lambda) \prod_{e \in E(\mathbf{C}_q)} (1 - P(c|e))}{g(c)} \quad (7)$$

where λ is the leak probability that the concept c is a related concept of c_q for no good reasons. The nominator is the probability that c is a concept of $E(c_q)$ and the denominator is a function $g(c)$ to penalize the extent containing noisy entities. By Noisy-Or model, any concept that is related to at least one entity in $E(c_q)$ will be captured.

In the above equations, we still need to estimate $P(c|e)$ (the probability that c is a concept of e) and $P(e|c)$ (the probability that e is an instance of c) as well as the prior probability of e and c . We use Probase data for the estimation as follows:

$$P(c|e) = \frac{n(c, e)}{n(e)}, P(e|c) = \frac{n(c, e)}{n(c)}, \quad (8)$$

where $n(c, e)$ is the number of co-occurrence between c and e , and $n(e)$ is the number of the occurrence of e in Probase. We set $P(e) \propto n(e)$ and $P(c) \propto n(c)$.

There are many functions that can be used to define $g(c)$ varying from different data quality of the knowledge base, and we use the following one:

$$g(c) = \frac{\delta + \sum_{e \in E(c), e \notin E_{\cup}(\mathbf{C}_q)} (n(e, c) + 1)}{\sum_{e \in E(c)} (n(e, c) + 1)} \quad (9)$$

where $E_{\cup}(\mathbf{C}_q) = \cup_{c' \in \mathbf{C}_q} E(c')$ and δ is small positive value less than 1. In the denominator, we count all entities that do not appear in instances of any shorter concept of c_q .

Entity Ordering After the instance-based expansion, we derive the relevance score $rel(c, \mathbf{C}_q)$. Given these scores, we rank all entries as follows:

$$rel(e, c_q) = \sum_c P(e|c) rel(c, \mathbf{C}_q) \quad (10)$$

The rationality of the ordering is obvious. The more relevant concepts contain an entity, the more likely the entity is a good answer. Next, we highlight two facts about the ordering. First, $rel(e, c_q)$ generates a linear ordering on all observed entities in the intersection as well as expanded entities. Although theoretically we cannot exclude the possibilities of ties, it is a rare case practically. Even tie happens, we can assign a random ordering to ensure the result ordering is linear. Second, the ordering is well defined on all concepts, including those in \mathbf{C}_q . In general, for a $c \in \mathbf{C}_q$, Eq 9 has a smallest value with the nominator as δ because c contains no entity outside of $E_{\cup}(\mathbf{C}_q)$. Hence, we assign almost no penalty to $c \in \mathbf{C}_q$ and consequently these concepts have a high relevance score. This is intuitively correct.

Seed Instances Generation

For the cases where $E(\mathbf{C}_q) = \emptyset$, our basic idea is to find a subset \mathbf{C}'_q of \mathbf{C}_q such that $E(\mathbf{C}'_q)$ is not empty. For each $E(\mathbf{C}'_q) \neq \emptyset$, we use the instance-based expansion to find new entities. Continue Example 1, all shorter concept subsets $\{c_1\}$, $\{c_2\}$, $\{c_3\}$, $\{c_1, c_2\}$, $\{c_1, c_3\}$, and $\{c_2, c_3\}$ are candidate \mathbf{C}'_q . Since we intersect less concepts, we have more chance to find a non-empty $E(\mathbf{C}'_q)$.

More formally, given a long concept c_q , let n be the number its modifiers, which means that $|\mathbf{C}_q| = n$. In our implementation, we exhaustively enumerate all \mathbf{C}'_q with capacity less than n . For most LCQs, we have $2^n - 2$ choices of \mathbf{C}'_q . In general, n is less than 10, hence the enumeration cost is acceptable.

Pairwise Ordering Constraint Note that the seed instance generation inherently implies a pairwise ordering constraint. Clearly, the more shorter concepts join the intersection, the more likely their concepts and entities expanded are related to the query concept. If e_1 has a higher order than e_2 , we denote it as $e_1 \succ e_2$. We formalize the constraint in Def 1. We illustrate the rationality of the constraint in Example 2. These constraints actually represent the partial order between entities.

Definition 1 (Pairwise Ordering Constraint) For any $e_i \in E(\mathbf{C}_q^{(1)})$ and $e_j \in E(\mathbf{C}_q^{(2)})$ such that $\mathbf{C}_q^{(1)} \subset \mathbf{C}_q$ and $\mathbf{C}_q^{(2)} \subset \mathbf{C}_q$, if $|\mathbf{C}_q^{(1)}| > |\mathbf{C}_q^{(2)}|$, we have $e_i \succ e_j$.

Example 2 Continue the running example, ‘great state school’ contains entities {virginia, unc, michigan, berkley} in Probase which overlaps only with c_1 and c_2 . Instead ‘Ivy League’ overlaps with c_1 , c_2 and c_3 . Hence, it is reasonable to believe that instances of ‘Ivy League’ are more likely to be the correct answers than that in ‘great state school’. In other words, we have the following constraint {harvard, princeton, ... \succ berkley, virginia, ...}

Ranking Aggregation

In this section, we present our solution to aggregate all ranking of entities under the constraints. We propose a probabilistic ranking optimization method based on Bradley-Terry model. This model quantifies the probability of generating a ranking permutation, and our goal is to identify the most likely permutation under the given constraints.

More specifically, we need to generate a better linear ordering by aggregating the baseline ranking (R_b) and ranking of entities after concept expansion (R_c) and the pairwise ordering constraints (R_p). We first derive our problem model by Bradley-Terry Model.

Bradley-Terry Model

Given a pair of individuals i and j , it estimates the probability that the pairwise comparison $i > j$ turns out true, as

$$P(i > j) = \frac{\pi_i}{\pi_i + \pi_j} \quad (11)$$

where π_i and π_j is a positive real-value score assigned to individual i and j . Our goal is to find an ideal scoring S that maximizes the probability of a permutation. The probability can be computed from both a global and a partial ordering constraint. Given S , s_x and s_y represent the ranking score of x and y , and $\pi_x = e^{s_x}$. Then, the probability that $x \succ y$ is:

$$P(x \succ y) = \frac{e^{s_x}}{e^{s_x} + e^{s_y}} \quad (12)$$

We now discuss how to formulate such probability for

- Total ordering $Z = \{Z_i\}_{i=1}^{n-1}$, where $Z_i = z_i \succ z_{i+1}, z_{i+2}, \dots, z_n$ as follows:

$$P(Z_i) = P(z_i \succ \{z_j | j > i\}) = \frac{e^{s_{z_i}}}{e^{s_{z_i}} + \sum_{j>i} e^{s_{z_j}}} \quad (13)$$

Then we denote $L(Z)$ as the probability to generalize the total ordering:

$$L(Z) = \log\left(\prod_{i=1}^{n-1} P(Z_i)\right) = \sum_{i=1}^{n-1} \log(P(Z_i)) \quad (14)$$

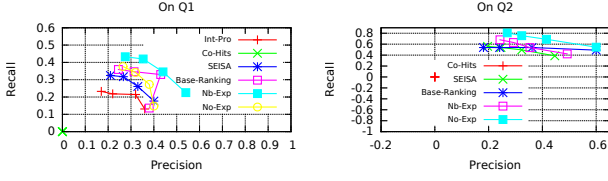


Figure 2: Precision and Recall curve @k

- Pairwise ordering between $X = \{x_i\}$ and $Y = \{y_j\}$, i.e., $X \succ Y$ as follows:

$$P(X \succ Y) = \frac{\sum_{x_i \in X} e^{s_{x_i}}}{\sum_{x_i \in X} e^{s_{x_i}} + \sum_{y_j \in Y} e^{s_{y_j}}} \quad (15)$$

Then the log likelihood of all partial order constraints in R_p , i.e., $L(R_p)$ can be computed as follows:

$$L(R_p) = \sum_{(X_i, Y_i) \in R_p} \log(P(X_i \succ Y_i)) \quad (16)$$

Thus, our goal is finding a scoring function S to maximize the probability obtained from total orderings (from baseline ranking R_b and entity ranking after concept expansion R_c) as well as the partial orders (from seed instance generation R_p). Hence, our ranking aggregation problem is:

$$\arg \max_S (1 - \alpha - \beta)L(R_b) + \alpha L(R_c) + \beta L(R_p) \quad (17)$$

where α, β are parameters used to tune the relative importance of each part. There are some strategies to tune the parameters, but they are out of the range of this paper. In our implementation, we just assign the same weight to all the parameters.

Model Computation

Since S (the scores of all entities) is a set of parameters which we need to learn, we use stochastic gradient descent (Bottou 2010) to compute the model. The gradient of every part of our optimization function are as follows:

$$\frac{\partial L(R_b)}{\partial s_i} = 1 - \frac{e^{s_i}}{\sum_{j=i} e^{s_j}}, \quad \frac{\partial L(R_c)}{\partial s_i} = 1 - \frac{e^{s_i}}{\sum_{j=i} e^{s_j}} \quad (18)$$

$$\frac{\partial L(R_p)}{\partial s_i} = \sum_{(X_j, Y_j) \in R_p, X_j \ni i} \left(\frac{e^{s_i} \sum_{k \in Y_j} e^{s_k}}{\sum_{k \in X_j} e^{s_k} (\sum_{k \in X_j} e^{s_k} + \sum_{k \in Y_j} e^{s_k})} \right) \quad (19)$$

Evaluation

In this section, we evaluate the precision and recall of our approaches with the comparison with the baselines and some of the state-of-the-art approaches.

Experimental Setup

We use Probase, which has a high coverage over entities and concepts, as our knowledge repository from which we retrieve answers for long concept query. To construct the long query set, we first find all concepts consisting of two words whose head occurs in more than four different concepts in Probase. Thus, for each head, we get many different modifiers. We randomly choose part of modifiers (from two to four) and combine them with the head to construct a long concept. We manually check each randomly generated long concept. If it is a meaningful concept, we accept it otherwise reject it.

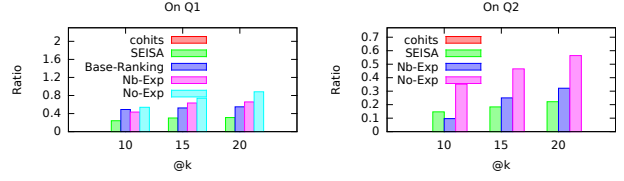


Figure 3: Ratio @k

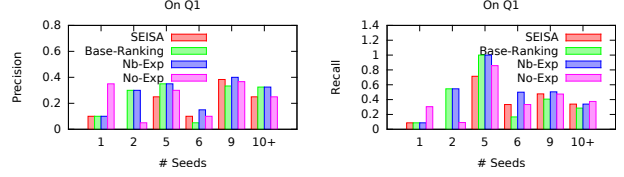


Figure 4: Average Precision and Average Recall @20 on Q_1

Dataset We generate two long query sets, a real set and a synthesized set respectively to evaluate our approach.

- **Real data set Q_1 .** Q_1 consists of 15 long concepts which we can find ground truth answer from Wikipedia directly. If the answer entity list exists in Wikipedia, we directly use it as the ground truth answer. Otherwise, we first find the entity lists of all shorter concepts in C_q and use their intersections as the ground truth answer. If any shorter concept has no entity list in Wikipedia, we just skip the long concept. By comparing the results generated by our solution with the ground truth answer, we can directly test the effectiveness of our solution.
- **Synthetic data set Q_2 .** Due to the data sparsity, entities in Probase may not overlap with some of the entities in Wikipedia and vice versa. This leads to a low recall and precision of our query answering. Hence, we further construct a synthesized query data set Q_2 . This query set consists of 20 long concepts. For these long concepts or their shorter concepts, we can not find the corresponding entity lists from Wikipedia. Yet the entity intersection of shorter concept is not empty in Probase. Then, we randomly eliminate half of the entities from the intersection and test whether our solution can find these deliberately removed entities.

Competitors We compare with the following solutions.

- **Decomposition-then-intersection (Int-Pro)** This is the high-precision baseline (Int-Pro for short) which we introduced in the introduction. In this baseline, we rank the results by their sum of co-occurrence frequency with the shorter concepts in C_q .
- **Co-Hits (PageRank)** Our problem can be seen as a problem of set expansion given the seed entities. Hits, PageRank and their variations have been adopted in some set expansion works such as SEAL (Wang and Cohen 2007). Since we can construct bipartite graph consisting of entities and concepts, the most appropriate one to compare is the generalized Co-Hits algorithm (Deng, Lyu, and King 2009).
- **SEISA** SEISA (He and Xin 2011) is a state-of-the-art set expansion algorithm using web lists. Here we implement its *Static Thresholding Algorithm* using the entity intersection of the shorter concepts of the query as the seed. Since we do not have a big web list corpus, we use Probase here instead.
- **Base-Ranking** This is the baseline ranking proposed in Section 4, which directly ranks entities without additional information from the expanded concepts.

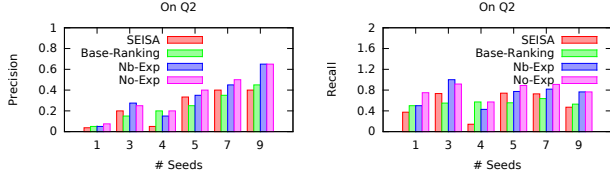


Figure 5: Average Precision and Average Recall @20 on Q_2

- **Nb-Exp** This is our approach using Naive-Bayes model to expand concepts.
- **No-Exp** This is our approach using Noisy-Or model to expand concepts.

Metrics We have two major components: the concept expansion part and the ranking aggregation part to evaluate. As for the concept expansion, our purpose is to evaluate how many new answer entities we can find to improve the recall. Let n^* be the number of entities appearing in the top- k answers but not in $\cap_{c \in C_q} e(c)$. Then we define $ratio@k$ to quantify the ability of our solution to find new entities:

$$ratio@k = \frac{n^*}{|\cap_{c \in C_q} e(c)| + 1} \quad (20)$$

To every query data set, we use average $ratio@k$ to evaluate the concept expansion part.

As for ranking aggregation, we report Precision@ k and Recall@ k . Let the number of the entities in the ranked list be n_r , and the number of the entities in the ground truth be n_g . Precision@ k is the number of the correct entities that are among the top- k in the ranked list divided by n_r . Recall@ k is the number of the correct entities that are among the top- k in the ranked list divided by n_g .

Results

Precision and Recall In Figure 2, we report precision and recall @5, @10, @15, and @20 (from left to right) of our solutions with the comparison with all competitors. On Q_2 , we find that our No-Exp approach always performs best both on precision and recall. It also shows that in this task Noisy-Or model is better than Naive-Bayes Model. Set expansion methods do not perform as well as our approaches in this data set. On Q_1 , we can see that the whole performance is not as good as the performance in the result of Q_2 . However, our Nb-Exp still performs the best on both precision and recall which reveals the effectiveness of our framework.

Influence of #seeds The number of entities in the intersection of the shorter concepts determines the performance of set expansion algorithm. We denote this number of #seeds. Hence, we evaluate precision and recall as the function of #seeds. The results on Q_1 and Q_2 are shown in Figure 4 and 5 respectively. We can see that on Q_2 , one of No-Exp and Nb-Exp always performs best on both precision and recall. When there is only a few seeds, set expansion algorithm is not a good choice. It is reasonable because there is not enough signal to expand concepts or entities.

Effectiveness of expansion Note that we propose concept expansion and seed instances generation to improve the recall. We find that our expansion efforts can find new entities

Table 1: Top Entities of Query: Top American Public University

Int-Pro	SEISA	No-Exp	GT from Wiki
berkeley	berkeley	berkeley	university of california berkeley
	cambridge	ucla	university of california los angeles
	louisville	michigan	university of virginia charlottesville
	harvard university	university of washington	university of michigan
	new york university	madison	university of north carolina chapel hill

Table 2: Top Entities of Query: Asian Arab Oil-exporting Country

Int-Pro	SEISA	No-Exp	GT from Wiki
dubai	kuwait	dubai	saudi arabia
iran	uae	saudi arabia	iraq
kuwait	bahrain	kuwait	uae
saudi arabia	saudi arabia	iraq	kuwait
uae	qatar	bahrain	qatar
	lebanon	oman	oman
	iran	qatar	bahrain

outside of entity intersections of short concepts in Figure 3. The results show that our No-Exp method can always find more new entities on both of the two data sets than other methods. Note that Base-Ranking is only a raking solution which can not find new entities and thus is not tested here.

Case Study

We further give case studies to show the rationality of our models and solutions. In Table 1 and Table 2, we show the results of the query ‘Top American Public University’ and ‘Asian Arab Oil-exporting Country’. We give results of our solution as well as the competitors including SEISA and Int-Pro. The results show that we have a much higher recall than Int-Pro, and higher precision and recall than SEISA. These suggest the effectiveness of our approach.

Conclusion

This paper studies the long concept query (LCQ), that a user provides a long concept consisting of a single head with more than one modifiers and obtains in return top- k entities which are the k most typical instances of the query. Such entity acquisition queries are increasingly important and common in people’s daily life. However, due to the limited concept coverage of the knowledge base and the data sparsity led by the patterns building the knowledge base, it is difficult to deal with this kind of query well, specially on recall. This paper provides a query processing method based on a conceptual taxonomy which aims to improve the recall of LCQ while maintaining a high precision. Our proposed method detects some semantically related concepts based on a probabilistic framework to populate the results based on a learning-based aggregation approach considering both entity orderings and pairwise ordering constraints. Our evaluation results with real data sets show that the results of the queries processed with this new method is significantly higher than that of existing solutions.

References

- [Ali and Meilă 2012] Ali, A., and Meilă, M. 2012. Experiments with kemeny ranking: What works when? *Mathematical Social Sciences* 64(1):28–40.
- [Betzler et al. 2010] Betzler, N.; Guo, J.; Komusiewicz, C.; and Niedermeier, R. 2010. Average parameterization and partial kernelization for computing medians. In *LATIN 2010: Theoretical Informatics*. Springer. 60–71.
- [Betzler, Brederbeck, and Niedermeier 2014] Betzler, N.; Brederbeck, R.; and Niedermeier, R. 2014. Theoretical and empirical evaluation of data reduction for exact kemeny rank aggregation. *Autonomous Agents and Multi-Agent Systems* 28(5):721–748.
- [Bottou 2010] Bottou, L. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer. 177–186.
- [Cohen-Boulakia, Denise, and Hamel 2011] Cohen-Boulakia, S.; Denise, A.; and Hamel, S. 2011. Using medians to generate consensus rankings for biological data. In *Scientific and Statistical Database Management*, 73–90. Springer.
- [Conitzer, Davenport, and Kalagnanam 2006] Conitzer, V.; Davenport, A.; and Kalagnanam, J. 2006. Improved bounds for computing kemeny rankings.
- [Davenport and Kalagnanam] Davenport, A., and Kalagnanam, J. A computational study of the kemeny rule for preference aggregation.
- [Deng, Lyu, and King 2009] Deng, H.; Lyu, M. R.; and King, I. 2009. A generalized co-hits algorithm and its application to bipartite graphs. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 239–248. ACM.
- [Ghahramani and Heller] Ghahramani, Z., and Heller, K. A. Bayesian sets.
- [He and Xin 2011] He, Y., and Xin, D. 2011. Seisa: set expansion by iterative similarity aggregation. In *Proceedings of the 20th international conference on World wide web*, 427–436. ACM.
- [Kleinberg 1999] Kleinberg, J. M. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)* 46(5):604–632.
- [Meila et al. 2012] Meila, M.; Phadnis, K.; Patterson, A.; and Bilmes, J. A. 2012. Consensus ranking under the exponential model. *arXiv preprint arXiv:1206.5265*.
- [Page et al. 1999] Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The pagerank citation ranking: bringing order to the web.
- [Pantel et al. 2009] Pantel, P.; Crestan, E.; Borkovsky, A.; Popescu, A.-M.; and Vyas, V. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, 938–947. Association for Computational Linguistics.
- [Sarmiento et al. 2007] Sarmiento, L.; Jijkuon, V.; de Rijke, M.; and Oliveira, E. 2007. More like these: growing entity classes from seeds. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, 959–962. ACM.
- [Wang and Cohen 2007] Wang, R. C., and Cohen, W. W. 2007. Language-independent set expansion of named entities using the web. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, 342–350. IEEE.
- [Wang and Cohen 2008] Wang, R. C., and Cohen, W. W. 2008. Iterative set expansion of named entities using the web. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 1091–1096. IEEE.
- [Wang et al. 2015] Wang, C.; Chakrabarti, K.; He, Y.; Ganjam, K.; Chen, Z.; and Bernstein, P. A. 2015. Concept expansion using web tables. In *Proceedings of the 24th International Conference on World Wide Web*, 1198–1208. International World Wide Web Conferences Steering Committee.
- [Wang, Wang, and Hu 2014] Wang, Z.; Wang, H.; and Hu, Z. 2014. Head, modifier, and constraint detection in short texts. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*, 280–291. IEEE.
- [Yin, Han, and Yu 2008] Yin, X.; Han, J.; and Yu, P. S. 2008. Truth discovery with multiple conflicting information providers on the web. *Knowledge and Data Engineering, IEEE Transactions on* 20(6):796–808.